EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	60	gregory near wright.in.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:08
S2	39	mario near wolczko.in.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:08
S3	15	matthew near seidl.in.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:09
S4	7356	"sun microsystems".as.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:09
S5	0	S4 and (de$1compil$4 and (virtual adj machine)).clm.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:10
S6	6	S4 and (de$1compil$4).clm.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:11
S7	282	S4 and (virtual adj machine).clm.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:11
S9	19	S4 and (method and optimiz$5 and (virtual adj machine)).clm.	US-PGPUB; USPAT	OR	ON	2007/05/25 10:12
S10	2341	717/146-148,151-161.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 10:15
S11	33	S10 and de$1compil$5	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 10:16
S12	15	S11 and (virtual adj machine)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 10:16
S13	443	S10 and (java adj virtual adj machine)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 10:20
S14	195	S13 and (native adj code)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 10:21

EAST Search History

S15	145	S14 and optimiz$5	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 10:22
S16	131	S15 and (@pd<"20030822" or @ad<"20030822" or @prad<"20030822" or @rlad<"20030822")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:32
S17	9	(US-6289506-$ or US-5999732-$ or US-6412107-$ or US-6412108-$ or US-6412109-$ or US-6513156-$ or US-6910205-$ or US-7032216-$ or US-7150012-$).did.	USPAT	OR	ON	2007/05/25 11:15
S18	5	S17 and (combin$4 or merg$4 or join$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:30
S19	110	de$1compil$5 and (java adj virtual adj machine)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:19
S20	43	S19 and intermediate	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:19
S21	40	S20 and (@pd<"20030822" or @ad<"20030822" or @prad<"20030822" or @rlad<"20030822")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:19
S22	1	S17 and de$1compil$5	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:31

S23	62	S19 and (byte$1code)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:32
S24	57	S23 and (@pd<"20030822" or @ad<"20030822" or @prad<"20030822" or @rlad<"20030822")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:32
S25	118954	(combin$4 or merg$4 or join$4) with (byte$1code or intermediate)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:52
S27	9263	combin$4 near3 (byte$1code or intermediate)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 11:52
S28	12	("5367685" | "5465258" | "5539907" | "5590331" | "5668999" | "5768592" | "5768593" | "5842017" | "5905895" | "5909577" | "5995754" | "6170083").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2007/05/25 14:38
S29	5	("5905895" | "5987256" | "6081665" | "6223202" | "6233733").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2007/05/25 14:56
S30	16709	combin$5 near3 (byte$1code or intermediate)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/25 15:00
S31	10	(US-6289506-$ or US-7032216-$ or US-7150012-$ or US-6910205-$ or US-6513156-$ or US-6412109-$ or US-6412108-$ or US-6412107-$ or US-5999732-$ or US-6151701-$).did.	USPAT	OR	ON	2007/05/26 13:19
S32	1	S31 and heap	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 13:19

EAST Search History

S33	7	(heap adj object) with (virtual adj machine)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 14:38
S36	21	call$1back with heap	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 14:38
S37	11	previous near3 (intermediate adj representation)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 15:06
S39	36244	"applicaiton binary interface" or abi	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 15:10
S41	185	("applicaiton binary interface" or abi) with (argument or parameter)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 15:13
S42	9	S41 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 15:13
S43	177	("applicaiton binary interface" or abi) with map$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 15:47
S44	1	S43 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/05/26 15:13

PORTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

"java virtual machine" optimization

SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **java virtual machine optimization**

Found **34,778** of **201,798**

Sort results by	relevance	Save results to a Binder
Display results	expanded form	Search Tips
		□ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale □ ▫ ▪ ■ ■

1 Sensor networks and performance analysis: Relative factors in performance analysis of Java virtual machines ■
Dayong Gu, Clark Verbrugge, Etienne M. Gagnon
June 2006 **Proceedings of the second international conference on Virtual execution environments VEE '06**
Publisher: ACM Press
Full text available: pdf(379.84 KB) Additional Information: full citation, abstract, references, index terms

Many new Java runtime optimizations report relatively small, single-digit performance improvements. On modern virtual and actual hardware, however, the performance impact of an optimization can be influenced by a variety of factors in the underlying systems. Using a case study of a new garbage collection optimization in two different Java virtual machines, we show the relative effects of issues that must be taken into consideration when claiming an improvement. We examine the specific and overal ...

Keywords: Java, caches, garbage collection, hardware counters, performance analysis

2 Applications I: A dynamic compiler for embedded Java virtual machines ■
Mourad Debbabi, Abdelouahed Gherbi, Lamia Ketari, Chamseddine Talhi, Nadia Tawbi, Hamdi Yahyaoui, Sami Zhioua
June 2004 **Proceedings of the 3rd international symposium on Principles and practice of programming in Java PPPJ '04**
Publisher: Trinity College Dublin
Full text available: pdf(195.32 KB) Additional Information: full citation, abstract, references, citings

A new acceleration technology for Java embedded virtual machines is presented in this paper. Based on the selective dynamic compilation technique, this technology addresses the J2ME/CLDC (Java 2 Micro Edition for Connected Limited Device Configuration) platform. The primary objective of our work is to come up with an efficient, lightweight and low-footprint accelerated embedded Java Virtual Machine. This is achieved by the means of integrating a selective dynamic compiler that we called E-Bunny ...

Keywords: J2ME/CLDC, Java, KVM, acceleration, embedded systems, performance, selective dynamic compilation, virtual machine

3 Embedded systems: applications, solutions and techniques (EMBS): Armed E-Bunny: a selective dynamic compiler for embedded Java virtual machine targeting ARM processors ■
Mourad Debbabi, Azzam Mourad, Nadia Tawbi
March 2005 **Proceedings of the 2005 ACM symposium on Applied computing SAC '05**

Publisher: ACM Press

Full text available: pdf(175.80 KB) Additional Information: full citation, abstract, references, citings

This paper presents a new selective dynamic compilation technique targeting ARM 16/32-bit embedded system processors. This compiler is built inside the J2ME/CLDC (Java 2 Micro Edition for Connected Limited Device Configuration) platform [8]. The primary objective of our work is to come up with an efficient, lightweight and low-footprint accelerated Java virtual machine ready to be executed on embedded machines. This is achieved by implementing a selective ARM dynamic compiler called Armed E-Bunn ...

Keywords: ARM, J2ME/CLDC, Java, KVM, embedded systems, selective dynamic compilation

4 A framework for optimizing Java using attributes

Patrice Pominville, Feng Qian, Raja Vallée-Rai, Laurie Hendren, Clark Verbrugge

November 2000 **Proceedings of the 2000 conference of the Centre for Advanced Studies on Collaborative research CASCON '00**

Publisher: IBM Press

Full text available: pdf(314.37 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents a framework for supporting the optimization of Java programs using attributes in Java class files. We show how class file attributes may be used to convey both optimization opportunities and profile information to a variety of Java virtual machines including ahead-of-time compilers and just-in-time compilers.We present our work in the context of Soot, a framework that supports the analysis and transformation of Java bytecode (class files)[21, 25, 26]. We demonstrate the frame ...

5 Effective management of multiple configurable units using dynamic optimization

Shiwen Hu, Madhavi Valluri, Lizy Kurian John

December 2006 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 3 Issue 4

Publisher: ACM Press

Full text available: pdf(443.24 KB) Additional Information: full citation, abstract, references, index terms

As one of the promising efforts to minimize the surging microprocessor power consumption, adaptive computing environments (ACEs), where microarchitectural resources can be dynamically tuned to match a program's run-time requirement and characteristics, are becoming increasingly common. In an ACE, efficient management of the configurable units (CUs) is vital for maximizing the benefit of resource adaptation. ACEs usually have multiple configurable hardware units, necessitating exploration of a la ...

Keywords: Adaptive computing environment (ACE), dynamic optimization, hotspots, power dissipation

6 Soot - a Java bytecode optimization framework

Raja Vallée-Rai, Phong Co, Etienne Gagnon, Laurie Hendren, Patrick Lam, Vijay Sundaresan

November 1999 **Proceedings of the 1999 conference of the Centre for Advanced Studies on Collaborative research CASCON '99**

Publisher: IBM Press

Full text available: pdf(79.70 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents Soot, a framework for optimizing Java bytecode. The framework is implemented in Java and supports three intermediate representations for representing Java bytecode: Baf, a streamlined representation of bytecode which is simple to manipulate; Jimple, a typed 3-address intermediate representation suitable for optimization; and Grimp, an aggregated version of Jimple suitable for decompilation. We describe the motivation for each representation, and the salient points in translat ...

7 Effective Adaptive Computing Environment Management via Dynamic Optimization
Shiwen Hu, Madhavi Valluri, Lizy Kurian John
March 2005 **Proceedings of the international symposium on Code generation and optimization CGO '05**
Publisher: IEEE Computer Society
Full text available: pdf(177.11 KB) Additional Information: full citation, abstract, citings, index terms

To minimize the surging power consumption of microprocessors, adaptive computing environments (ACEs) where microarchitectural resources can be dynamically tuned to match a program's runtime requirement and characteristics are becoming increasingly common. Adaptive computing environments usually have multiple configurable hardware units, necessitating exploration of a large number of combinatorial configurations in order to identify the most energy-efficient configuration. In this paper, we propo ...

8 Practicing JUDO: Java under dynamic optimizations
Michał Cierniak, Guei-Yuan Lueh, James M. Stichnoth
May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation PLDI '00**, Volume 35 Issue 5
Publisher: ACM Press
Full text available: pdf(190.06 KB) Additional Information: full citation, abstract, references, citings, index terms

A high-performance implementation of a Java Virtual Machine (JVM) consists of efficient implementation of Just-In-Time (JIT) compilation, exception handling, synchronization mechanism, and garbage collection (GC). These components are tightly coupled to achieve high performance. In this paper, we present some static anddynamic techniques implemented in the JIT compilation and exception handling of the Microprocessor Research Lab Virtual Machine (MRL VM), ...

9 Diverse topics: Field level analysis for heap space optimization in embedded java environments
Guangyu Chen, Mahmut Kandemir, N. Vijaykrishnan, Mary Janie Irwin
October 2004 **Proceedings of the 4th international symposium on Memory management ISMM '04**
Publisher: ACM Press
Full text available: pdf(1.67 MB) Additional Information: full citation, abstract, references, index terms

Memory constraint presents one of the critical challenges for embedded software writers. While circuit-level solutions based on cramming as many bits as possible into the smallest area possible are certainly important, memory-conscious software can bring much higher benefits. Focusing on an embedded Java-based environment, this paper studies potential benefits and challenges when heap memory is managed at a field granularity instead of object. This paper discusses these benefits and challenge ...

Keywords: garbage collection, java virtual machine

10 The Performance of Runtime Data Cache Prefetching in a Dynamic Optimization System
Jiwei Lu, Howard Chen, Rao Fu, Wei-Chung Hsu, Bobbie Othmer, Pen-Chung Yew, Dong-Yuan Chen
December 2003 **Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture MICRO 36**
Publisher: IEEE Computer Society
Full text available: pdf(253.79 KB) Additional Information: full citation, abstract, citings, index terms

Traditional software controlled data cache prefetching isoften ineffective due to the lack of runtime cache miss andmiss address information. To overcome this limitation, weimplement runtime data cache prefetching in the dynamicoptimization system ADORE

(ADaptive Object code RE-optimization).Its performance has been compared withstatic software prefetching on the SPEC2000 benchmarksuite. Runtime cache prefetching shows better performance.On an Itanium 2 based Linux workstation, it can increasepe ...

11 Dynamic Adaptive compilation: An infrastructure for adaptive dynamic optimization
Derek Bruening, Timothy Garnett, Saman Amarasinghe
March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**
Publisher: IEEE Computer Society
Full text available: pdf(1.16 MB) Additional Information: full citation, abstract, references, citings, index terms

Dynamic optimization is emerging as a promising approach to overcome many of the obstacles of traditional static compilation. But while there are a number of compiler infrastructures for developing static optimizations, there are very few for developing dynamic optimizations. We present a framework for implementing dynamic analyses and optimizations. We provide an interface for building external modules, or clients, for the DynamoRIO dynamic code modification system. This interface abstracts awa ...

12 HBench:Java: an application-specific benchmarking framework for Java virtual machines
Xiaolan Zhang, Margo Seltzer
June 2000 **Proceedings of the ACM 2000 conference on Java Grande JAVA '00**
Publisher: ACM Press
Full text available: pdf(775.76 KB) Additional Information: full citation, references, citings, index terms

Keywords: Java performance, benchmarking

13 Hardware implementation 2: JIT compiler optimizations for stack-based processors in embedded platforms
Giuseppe Di Giore, Antonella Di Stefano, Giovanni Morana, Corrado Santoro
October 2006 **Proceedings of the 4th international workshop on Java technologies for real-time and embedded systems JTRES '06**
Publisher: ACM Press
Full text available: pdf(453.16 KB) Additional Information: full citation, abstract, references, index terms

This paper describes the optimizations introduced in porting the CDCHI virtual machine to an ST20-based platform for set-top-boxes. ST20 is a VL-RISC processor by STMicroelectronics featured by a stack-based architecture; this characteristic suggests an easy porting of a Java virtual machine and the associated JIT compiler. However, since the stack of ST20 is very limited (it has only three locations), proper techniques are needed in order to avoid frequent (and heavy) "spill" operations, i.e. s ...

Keywords: CDCHI, JIT compilers, ST20, embedded systems, set-top-boxes

14 Applications I: a synergy between efficient interpretation and fast selective dynamic compilation for the acceleration of embedded Java virtual machines
Mourad Debbabi, Abdelouahed Gherbi, Lamia Ketari, Chamseddine Talhi, Hamdi Yahyaoui, Sami Zhioua
June 2004 **Proceedings of the 3rd international symposium on Principles and practice of programming in Java PPPJ '04**
Publisher: Trinity College Dublin
Full text available: pdf(166.34 KB) Additional Information: full citation, abstract, references

We propose, in this paper, a technique for the acceleration of embedded Java virtual machines. The technique relies on an established synergy between efficient interpretation and selective dynamic compilation. Actually, efficient interpretation is achieved by a generated threaded interpreter that is made of a pool of codelets. The latter are native code units efficiently implementing the dynamic semantics of a given bytecode. Besides, each codelet carries out the dispatch to the next bytecode el ...

Keywords: J2ME/CLDC, Java, KVM, acceleration, code reuse, embedded systems, performance, selective dynamic compilation, threaded interpretation, virtual machine

15 Compilation and dynamic optimization: Instruction folding in a hardware-translation based java virtual machine
Hitoshi Oi
May 2006 **Proceedings of the 3rd conference on Computing frontiers CF '06**
Publisher: ACM Press
Full text available: pdf(169.56 KB) Additional Information: full citation, abstract, references, index terms

Bytecode hardware-translation improves the performance of a Java Virtual Machine (JVM) with small hardware resource and complexity overhead. Instruction folding is a technique to further improve the performance of a JVM by reducing the redundancy in the stack-based instruction execution. However, the variable instruction length of the Java bytecode makes the folding logic complex. In this paper, we propose a folding scheme with reduced hardware complexity and evaluate its performance. For seven ...

Keywords: hardware-translation, instruction folding, java virtual machine, performance evaluation

16 Dynamic native optimization of interpreters
Gregory T. Sullivan, Derek L. Bruening, Iris Baron, Timothy Garnett, Saman Amarasinghe
June 2003 **Proceedings of the 2003 workshop on Interpreters, virtual machines and emulators IVME '03**
Publisher: ACM Press
Full text available: pdf(150.25 KB) Additional Information: full citation, abstract, references, citings, index terms

For domain specific languages, "scripting languages", dynamic languages, and for virtual machine-based languages, the most straightforward implementation strategy is to write an interpreter. A simple interpreter consists of a loop that fetches the next bytecode, dispatches to the routine handling that bytecode, then loops. There are many ways to improve upon this simple mechanism, but as long as the execution of the program is driven by a representation of the program other than as a stream of n ...

17 A dynamic optimization framework for a Java just-in-time compiler
Toshio Suganuma, Toshiaki Yasue, Motohiro Kawahito, Hideaki Komatsu, Toshio Nakatani
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications OOPSLA '01**, Volume 36 Issue 11
Publisher: ACM Press
Full text available: pdf(2.12 MB) Additional Information: full citation, abstract, references, citings, index terms

The high performance implementation of Java Virtual Machines (JVM) and just-in-time (JIT) compilers is directed toward adaptive compilation optimizations on the basis of online runtime profile information. This paper describes the design and implementation of a dynamic optimization framework in a production-level Java JIT compiler. Our approach is to employ a mixed mode interpreter and a three level optimizing compiler, supporting quick, full, and special optimization, each of which has a differ ...

18
System on chip design and software supports (SODSS): Real-time Java processor optimized for RTSJ

Zhilei Chai, Wenke Zhao, Wenbo Xu
March 2007 **Proceedings of the 2007 ACM symposium on Applied computing SAC '07**
Publisher: ACM Press
Full text available: pdf(101.40 KB) Additional Information: full citation, abstract, references, index terms

Due to the preeminent work of the real-time specification for Java(RTSJ), Java is increasingly expected to become the leading programming language in real-time systems. To provide a Java platform suitable for real-time applications, a real-time Java processor can execute Java bytecode directly is proposed in this paper. This processor provides efficient support in hardware for mechanisms specified in the RTSJ and offers a simpler programming model through ameliorating the scoped memory of the ...

Keywords: Java Virtual Machine (JVM), Real-time Java platform, Real-time Java processor, Real-time specification for Java (RTSJ), Worst Case Execution Time (WCET)

19 Speculative execution: A new speculation technique to optimize floating-point performance while preserving bit-by-bit reproducibility
Mikio Takeuchi, Hideaki Komatsu, Toshio Nakatani
June 2003 **Proceedings of the 17th annual international conference on Supercomputing ICS '03**
Publisher: ACM Press
Full text available: pdf(227.01 KB) Additional Information: full citation, abstract, references, index terms

The bit-by-bit reproducibility of floating-point results, which is defined by the IEEE 754 standard, prohibits optimizations such as reassociation and the use of native operations such as fused multiply-add (FMA), and thus it significantly impairs floating-point performance. Recent network-oriented languages such as Java strictly conform to the standard, and thus their numerical computing performance becomes inherently lower than conventional languages.In this paper, we propose a new software te ...

Keywords: IA-64, IEEE 754, Java, accuracy, bit-by-bit reproducibility, floating-point speculation, fused multiply-add, instruction-level parallelism, just-in-time compiler, loop unrolling, prefetching, privatization, reassociation, software pipelining, striding

20 Tuning garbage collection for reducing memory system energy in an embedded java environment
G. Chen, R. Shetty, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, M. Wolczko
November 2002 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 1 Issue 1
Publisher: ACM Press
Full text available: pdf(740.23 KB) Additional Information: full citation, abstract, references, citings, index terms

Java has been widely adopted as one of the software platforms for the seamless integration of diverse computing devices. Over the last year, there has been great momentum in adopting Java technology in devices such as cellphones, PDAs, and pagers where optimizing energy consumption is critical. Since, traditionally, the Java virtual machine (JVM), the cornerstone of Java technology, is tuned for performance, taking into account energy consumption requires reevaluation, and possibly redesign of t ...

Keywords: Garbage collector, Java Virtual Machine (JVM), K Virtual Machine (KVM), low power computing

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player